Improving Entity Linking by Encoding Type Information into Entity Embeddings

Tianran Li, Erguang Yang, Yujie Zhang[†], Jinan Xu, Yufeng Chen
School of Computer and Information Technology, Beijing Jiaotong University,
Beijing 100044, China
[†]yjzhang@bjtu.edu.cn

Abstract

Entity Linking (EL) refers to the task of linking entity mentions in the text to the correct entities in the Knowledge Base (KB) in which entity embeddings play a vital and challenging role because of the subtle differences between entities. However, existing pre-trained entity embeddings only learn the underlying semantic information in texts, yet the fine-grained entity type information is ignored, which causes the type of the linked entity is incompatible with the mention context. In order to solve this problem, we propose to encode fine-grained type information into entity embeddings. We firstly pre-train word vectors to inject type information by embedding words and fine-grained entity types into the same vector space. Then we retrain entity embeddings with word vectors containing fine-grained type information. By applying our entity embeddings to two existing EL models, our method respectively achieves 0.82% and 0.42% improvement on average F1 score of the test sets. Meanwhile, our method is model-irrelevant, which means it can help other EL models.

1 Introduction

Entity Linking (EL) refers to the task of linking entity mentions in the text to correct entities in the Knowledge Base (KB) to help text comprehension by making use of rich semantic information in KB. The state-of-the-art entity linking models ignore fine-grained entity type information which causes the type of the linked entity is incompatible with the mention context. For example, among the error cases in AIDA-CoNLL(Hoffart et al., 2011) dataset produced by Le and Titov (2018), the mention "Japan" in the sentence "Japan began the defense of their Asian Cup" is linked to the entity "Japan"*(country)* while it should be linked to the entity "Japan national football team"*(sports team)*. And such errors account for more than 1/2 of the total errors which show that their model cannot correctly distinguish different types of candidate entities.

Entity embeddings are especially important to entity linking task for the sake of distinguishing entities with similar characters. We hope the entity embeddings can have type information so as to differentiate entities in terms of categories. However, previous pre-trained entity embeddings are learned by using plenty of texts, which ignores fine-grained type information (Yamada et al., 2016; Ganea and Hofmann, 2017). As a result, the similar entity vectors in the vector space are of different types. In order to alleviate this problem, we firstly pre-train word vectors to inject type information by embedding words and fine-grained entity types into the same vector space. Then, we follow Ganea and Hofmann (2017) to retrain the representation of each entity by making it closer to the word vectors of its context, using the word vectors containing type information. Finally, we apply the entity embeddings into the existing EL models without modifying the model architectures. Compared with two baseline models, our method respectively achieves 0.82% and 0.42% improvement on average F1 score of the test sets and obtains new state-of-the-art results on multiple datasets. Besides, detailed experimental analysis confirms the hypothesis that encoding type information in entity embeddings can reduce the type error cases. Our contributions can be summarized as follows:

Proceedings of the 20th China National Conference on Computational Linguistics, pages 1087-1095, Hohhot, China, Augest 13 - 15, 2021.
(c) Technical Committee on Computational Linguistics, Chinese Information Processing Society of China

- We propose a simple but effective method to encode fine-grained entity type information into entity embeddings. Moreover, our method does not need to change the model structure thus can help other entity linking models.
- Experiment results on the EL task show that our method achieves substantial improvements and obtains new state-of-the-art results on multiple datasets.
- Detailed experimental analysis shows that introducing type information can correct substantial type error cases produced by the baseline model.

2 Background and Related Works

2.1 local and global entity linking models

The local entity linking model disambiguates each mention separately and measures the relevance scores between each candidate entity and mention context designed by Ganea and Hofmann (2017):

$$\Psi(e_i, c_i) = \mathbf{x}_{e_i}^T \mathbf{B} f(c_i) \tag{1}$$

where **B** is a learnable diagonal matrix, $\mathbf{x}_{e_i}$ is the embedding of entity e_i, $f(c_i)$ applies hard attention mechanism to obtain the feature representation of c_i which is the local context around mention m_i.

The global entity linking model encourages all entities in a document to be consistent with the topic. Ganea and Hofmann (2017) define a pairwise global score by calculating the consistency between each pair of entities in document D:

$$\Phi(e_i, e_j|D) = \frac{1}{n-1} \mathbf{x}_{e_i}^T \mathbf{C} \mathbf{x}_{e_j} \tag{2}$$

where **C** is a diagonal matrix, *n* is the number of mentions in document D. Based on Formula 2, Le and Titov (2018) propose a pairwise global score considering K potential relations as follows:

$$\Phi(e_i, e_j|D) = \sum_{k=1}^{K} \alpha_{ijk} \mathbf{x}_{e_i}^T \mathbf{R}_k \mathbf{x}_{e_j} \tag{3}$$

where $\mathbf{R}_k$ is a diagonal matrix to measure potential relationship k between entities e_i and e_j, and α_{ijk} is the weight corresponding to relation k.

2.2 Related Works

Entity Embeddings: Entity embeddings encode the structural or textual information of candidate entities. According to the different learning contents, the existing pre-trained entity embeddings are mainly categorized into the following three types:

(1) Context-based. Embeddings are typically trained using canonical Wikipedia articles that contain a lot of contextual information (Yamada et al., 2016; Ganea and Hofmann, 2017; Eshel et al., 2017; Rijhwani et al., 2019).

(2) Graph-based. This method builds entity graph based on KB and generates graph embeddings to capture structured information (Cao et al., 2018; Sevgili et al., 2019; Zhou et al., 2020).

(3) Description-based. This approach makes use of descriptions, names, redirects of entities in Wikipedia, which are dictionary-based data (Gupta et al., 2017; Logeswaran et al., 2019; Chen et al., 2020; Hou et al., 2020).

Our approach is similar to Hou et al.(2020) that generates semantic type words chosen from the first sentence of entity description pages in Wikipedia and combine them with entity embeddings of Ganea and Hofmann (2017) through linear aggregation. Our approach is also similar to Chen et al.(2020) that uses the pre-trained BERT (Devlin et al., 2018) to capture the latent type information in the mention context. Compared with their methods, the main advantage of our approach is that we explicitly encode fine-grained type information into the entity embeddings, while they implicitly capture the underlying type information.

Proceedings of the 20th China National Conference on Computational Linguistics, pages 1087-1095, Hohhot, China, Augest 13 - 15, 2021.

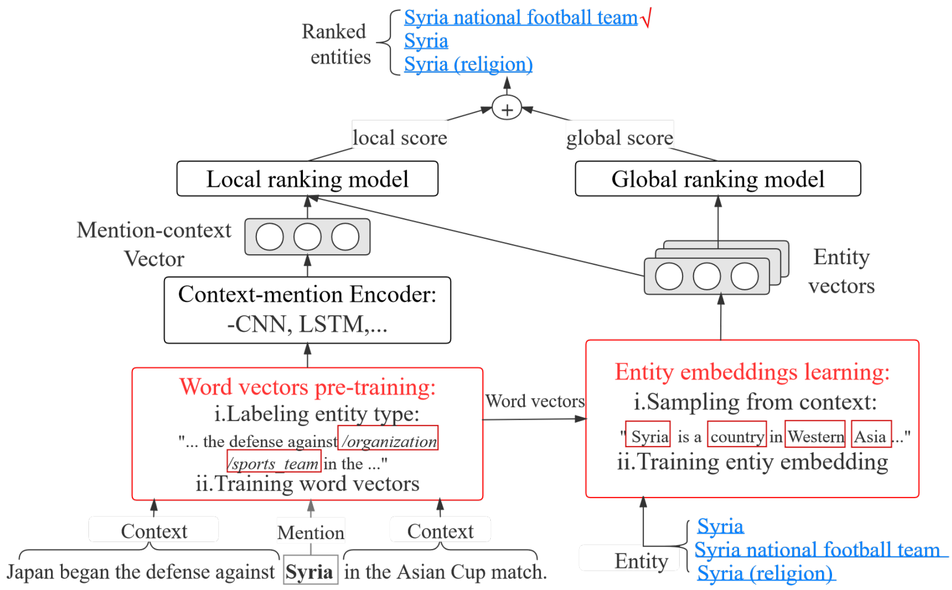

Figure 1: Overall framework of our method. The main component of our method is painted in red. We first pre-train word vectors to encode entity type information using Wikipedia articles with entity type tags. Then we utilize the word vectors with type information to train entity embedding. Both of the local and global ranking model can leverage the type information in entity embeddings and word vectors.

Fine-grained Entity Type: Existing entity type labeling systems classify entities in texts into fine-grained types. Our work utilizes ZOE (Zhou et al., 2018) system, which predicts the type in the FIGER (Ling and Weld, 2012) type set containing 112 two-dimensional entity types, such as *"/location/country"* and *"/location/city"*.

3 Our Method

Our method mainly consists of four steps: (1) Labeling named entities in Wikipedia articles as corresponding types. (2) Pre-training word and entity type embeddings by employing Skip-gram model (Mikolov et al., 2013). (3) Training the representation for each entity using the word vectors obtained in the previous step. (4) Combining entity vectors trained by Genea and Hofmann (2017) (hereafter referred to as "Wikitext entity embeddings") and our entity embeddings. The word vectors obtained in step (2) are also fused with the word vectors they use in the same way. Finally, we apply the result word and entity embeddings to the existing entity linking models. The overall architecture of our method is illustrated in Figure 1.

3.1 Entity Labeling

To embed words and entity types into the same vector space, we adopt a simple but effective preprocessing method: labeling named entities in Wikipedia articles as corresponding fine-grained 2-dimensional entity types in the FIGER type set as shown in Figure 2. We believe that the same type entities tend to appear in

(a)**Beaverton** is a community in **Brock Township** in the **Regional Municipality of Durham** , **Ontario** , **Canada**.
(b) */location/city* is a community in */location/city* in the */location* , */location/province* , */location/country*.

Figure 2: Example of preprocessed sentences.

Proceedings of the 20th China National Conference on Computational Linguistics, pages 1087-1095, Hohhot, China, Augest 13 - 15, 2021.

the similar contexts. To train embedding for each entity type, we utilize quite a few contexts of entities with the same type. For example, the embedding of "/location/country" would be trained from the contexts of all the country entities in Wikipedia articles. Specifically, we utilize hyperlinks in annotated Wikipedia corpus to extract the entity mention and map Freebase type of each entity mention into a set of fine-grained types.

3.2 Word and Type Vectors Pre-training

After processing the Wikipedia document, we employ the Skip-gram model (Mikolov et al., 2013) to learn uncased 300-dimensional embeddings for both words and entity types with a minimum word frequency cut-off of 5 and a window size of 5. We project 1200 words vectors of 6 different types and corresponding type vectors using the T-SNE algorithm (Maaten, 2014), as shown in Figure 3. It can be observed that the word vectors are clustered according to the fine-grained entity type. For instance, "mouse", "monkey" and "dog" are close to the type */livingthing/animal*. These results indicate our method can encode the type information into word vectors. We also observe that there are some overlaps between the entities of type */location/country* and */language* which is possibly due to the similar contexts or annotation errors.

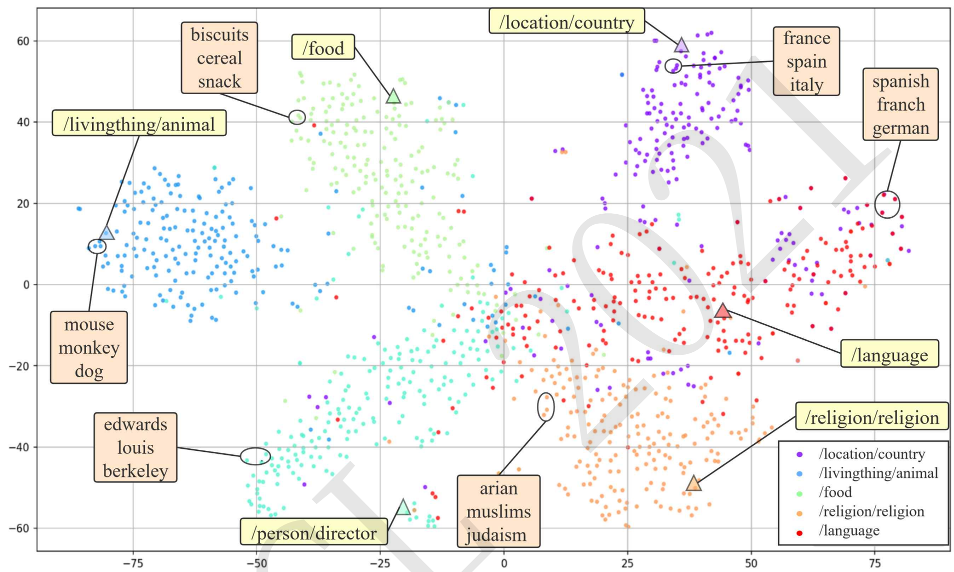

Figure 3: Two-dimensional representation of the vector space containing word and entity types embeddings. Type embeddings are represented by triangle, while words are represented by dot, and color represents the type to which they belong.

3.3 Entity Embeddings Training

We utilize the distribution hypothesis to learn the entity embeddings, where we use the word vectors obtained in Section 3.2 as the input. During training, we sample 20 context words around each entity as positive samples and 5 words at random as negative samples each iteration. Thus we train the entity embedding by making it close to the context words and further from other words in the vector space. Finally, the embeddings are normalized to obtain the joint distribution of entities and words on the unit sphere:

$$J(\mathbf{z}; e) = E_{\omega^+|e} E_{\omega^-}[h(\mathbf{z}; \omega^+, \omega^-)] \tag{4}$$

$$h(\mathbf{z}; w, v) = [\gamma - < \mathbf{z}, x_w - x_v >]_+ \tag{5}$$

$$x_e = \arg\min_{\mathbf{z}:||\mathbf{z}||=1} J(\mathbf{z}; e) \tag{6}$$

where x_e is the final entity embedding, γ is a margin parameter, $<, >$ denotes dot product, $[.]_+$ denotes the RELU function, (ω^+, ω^-) denotes the sample positive word and negative word respectively, E denotes the

Proceedings of the 20th China National Conference on Computational Linguistics, pages 1087-1095, Hohhot, China, Augest 13 - 15, 2021.

the process of sampling word vectors according to the given entity, **z** is the entity representation. The main idea is to find a **z** on the unit sphere that is closer to the positive samples and farther from the negative samples.

3.4 Entity Embeddings Fusing

In order to exploit the advantages of different entity embeddings, we combine our entity embeddings with Wikitext entity embeddings for entity linking, and the word vectors are fused in the same way. Specifically, we adopt three fusion methods:

i) Linearly Interpolation. Similar to Hou et al.(2020), we use a parameter α to control the weight of the two entity embeddings.

$$\mathbf{e}^{\alpha} = (1-\alpha)\mathbf{e}^{\omega} + \alpha\mathbf{e}^{t} \tag{7}$$

where $\mathbf{e}^{\omega}$ is the Wikitext entity embedding, $\mathbf{e}^{t}$ is our embedding.

ii) Nonlinear Interpolation. We also use a parameter β to control the weight of the two entity embeddings.

$$\mathbf{e}^{n} = (1-\beta) * \mathbf{e}^{\omega} + \beta * Tanh(\mathbf{e}^{t}) \tag{8}$$

iii) Gate Interpolation.

$$r, z = \sigma(W[\mathbf{e}^{\omega}; \mathbf{e}^{t}]) \tag{9}$$

$$\mathbf{e}^{g} = r \cdot \mathbf{e}^{\omega} + z \cdot \mathbf{e}^{t} \tag{10}$$

where W is a learnable matrix and σ is the sigmoid function, [;] denotes the concatenation process.

4 Experiments

4.1 Datasets and Evaluation Metric

We use standard benchmark dataset AIDA-train in AIDA-CoNLL (Hoffart et al., 2011) for training, AIDA-A for validating, AIDA-B for testing. We also use MSNBC, AQUQINT, ACE2004, WNED-WIKI (WIKI), WNED-CWEB (CWEB) which come from different domains for evaluation just like Ganea and Hofmann (2017) and Le and Titov (2018; 2019). It should be noted that CWEB and WIKI are believed to be less reliable (Ganea and Hofmann, 2017). The standard micro-F1 score is employed as evaluation metric.

4.2 Experimental Settings

We use canonical Wikipedia dumps data[0] published on 2020-07-01 which consists of 1.3G tokens to train word and type vectors. We totally annotate 157.4M entity mentions with their types. We use the method of Section 3.3 to train our entity vectors on the entity similarity task shared by Genea and Hofmann (2017), and finally we obtain 276,030 entity vectors containing type information according to the entity vocabulary.

The parameter α in Equation 7 and β in Equation 8 greatly affect the experimental results, we present the best group of results. To demonstrate the effectiveness of our entity embeddings, we use the existing state-of-the-art entity linking models **mulrel** (Le and Titov, 2018) (relations number K = 3) and **wnel** (Le and Titov, 2019) as our baseline models. To make our results comparable, we just replace the entity embeddings (Ganea and Hofmann, 2017) and the word vectors (Mikolov et al., 2013) that they used without modifying their model architectures. Besides, we follow Ganea and Hofmann (2017) to only consider the in-KB mentions and use the same candidate generation strategy. Similar to Le and Titov (2018), we run each combination of entity embedding and linking model for 5 times, record the mean and 95% confidence intervals of micro F1 score.

[0]https://dumps.wikimedia.org/enwiki/latest/

Proceedings of the 20th China National Conference on Computational Linguistics, pages 1087-1095, Hohhot, China, Augest 13 - 15, 2021.
(c) Technical Committee on Computational Linguistics, Chinese Information Processing Society of China

Entity Embeddings	Models	AIDA-B	MSNBC	AQUAINT	ACE2004	CWEB	WIKI	Avg
Wiki + Unlabelled documents								
-	Plato	86.4	-	-	-	-	-	-
G and H (2017)	wnel	89.66±0.16	92.2 ±0.2	90.7±0.2	88.1±0.0	78.2±0.2	81.7±0.1	86.18
Hou et al.(2020)	wnel	89.23±0.31	92.15±0.24	91.22±0.18	88.02±0.15	78.29±0.17	**81.92±0.36**	86.32
Ours(α = 0.1)	wnel	89.44±0.13	**92.46±0.08**	**91.69±0.10**	**88.13±0.00**	**78.90±0.05**	80.82±0.44	**86.39**
Ours(β = 0.1)	wnel	**90.40±0.14**	**92.49±0.01**	**91.64±0.15**	**88.21±0.22**	**78.77±0.01**	81.90±0.15	**86.60**
Ours(*gate*)	wnel	89.28±0.12	**92.55±0.08**	**91.58±0.15**	87.89±0.27	**78.77±0.06**	81.10±0.44	**86.38**
Fully-supervised (Wiki+ AIDA train)								
Yamada et al. (2016)	JEWE	91.5	-	-	-	-	-	-
G and H (2017)	Deep-ed	92.22±0.14	93.7±0.1	88.5±0.4	88.5±0.3	77.9±0.1	77.5±0.1	85.22
G and H (2017)	mulnel	93.07±0.27	93.9±0.2	88.3±0.6	89.9±0.8	77.5±0.1	78.0±0.1	85.5
G and H (2017)	DCA	**93.73±0.2**	93.8±0.0	88.25±0.4	90.14±0.0	75.59±0.3	78.84±0.2	85.32
Chen et al. (2020)	BERT-Sim	93.54±0.12	93.4±0.1	89.8±0.4	88.9±0.7	77.9±0.4	**80.1±0.4**	86.02
Hou et al. (2020)	mulnel	92.63±0.14	94.26±0.17	88.47±0.23	90.7±0.28	77.41±0.21	77.66±0.23	85.7
Ours(α = 0.1)	mulnel	92.41±0.32	93.59±0.10	**90.57±0.45**	90.22±0.82	77.86±0.26	77.94±0.60	**86.04**
Ours(β = 0.1)	mulnel	92.85±0.13	93.96±0.27	**90.15±0.38**	**90.87±0.65**	**78.07±0.27**	78.67±0.27	**86.32**
Ours(*gate*)	mulnel	92.97±0.07	**94.29±0.35**	**90.10±0.35**	90.30±0.44	77.80±0.06	77.83±0.40	**86.06**

Table 1: Comparison of experimental results of different models. The last column is the average of F1 scores on the five test sets. Underlined scores show cases where we outperforms the baseline models **wnel** and **mulnel**.

4.3 Results

As shown in Table 1,we compare the experimental results on the six test sets with several state-of-the-art models: Plato (Lazic et al., 2015), JEWE (Yamada et al., 2016), Deep-ed (Ganea and Hofmann, 2017), DCA (Yang et al., 2019), BERT-Sim (Chen et al., 2020) etc. The models are divided into two groups: (1) models using Wikipedia and Unlabelled document which are weakly-supervised; (2) models using Wikipedia and labelled dataset AIDA-train (Hoffart et al., 2011) which are fully-supervised.

For the **mulrel** model, we achieve new state-of-the-art results on MSNBC, AQUAINT, ACE2004 and CWEB datasets and the average F1 score of the five out-domain test sets. Compared to the baseline model, both of our approaches using nonlinear interpolation and gate interpolation can improve performance on four of the six datasets. For the **wnel** model, our entity embeddings obtain new state-of-the-art results on five of the six test sets, and obtain the optimal average F1 score of the five out-domain test sets. The experimental results can further draw the following three conclusions:

(1) In the weakly-supervised setting, our method achieves 0.74% performance improvement on the in-domain dataset and the improvement of the out-domain test sets is relatively solid. It can be seen that our embeddings perform better in the scenario with low resources.

(2) In most cases, three fusion methods can improve multiple datasets which come from different domains. And we respectively achieve 0.82% and 0.42% improvement on average F1 score of the five test sets. This demonstrates our method performs well on cross-domain issues.

(3) For different fusion ways of entity embeddings, nonlinear interpolation reveals excellent and solid results in both of the fully-supervised and weakly-supervised settings, gate interpolation and linearly interpolation also produce competitive results. Comparatively speaking, nonlinear interpolation is the most appropriate to combine the advantages of the two kinds of embeddings.

Compared with the entity embeddings of Hou et al. (2020) and Ganea and Hofmann (2017), our embeddings achieve solid performance improvement on most of the datasets, proving the effectiveness of the introduction of type information. Compared with the BERT-based entity embeddings of Chen et al. (2020), we achieve 0.32% performance improvement on five out-domain datasets using non-linear interpolation, while we achieve lower performance on the in-domain dataset AIDA. This experimental result also shows the advantages of making use of the latest pre-trained language models which have powerful encoding capability.

Contexts	**Mulrel** or **Wnel**	**Our model**
(1) **Palestinians** await word from Israel.	Palestine (*/location/country*)	Palestinian people (*/people/ethnicity*)
(2)The victims of the poisoning were in stable condition, but three suffered mild **liver disorders**.	Liver (*/body part*)	Liver disease (*/disease*)
(3) That would allow it to use clean burning **hydrogen** to generate electricity.	Hydrogen vehicle (*/product*)	Hydrogen (*/chemistry*)
(4) ...Who has joined Espanyol of Barcelona from **Nantes** since missing the European championship finals through injury.	Nantes (*/location/city*)	FC Nantes (*/organization/sports league*)

Table 2: Comparison of results between beseline model and our model. Mentions in texts are bold, the type of each entity is italic.

Methods	George Washington (*/person/politician*)	Beijing (*/location/city*)	On the Origin of Species (*/art/written work*)
Ganea and Hofmann (2017)	Abraham lincoln **Gilbert Du Motier,_Marquis de Lafayette** **American revolutionary war**	Seoul Shanghai **China**	**Charles Darwin** **Evolution** **Thomas Henry Huxley**
Ours	Abraham lincoln Ulysses S. Grant Jefferson Davis	Athens Seoul tokyo	History of evolutionary thought Introduction to evolution Theistic evolution

Table 3: Examples of nearest entities in Ganea and Hofmann(2017) and our entity representation space. The entity whose type is inconsistent with the entity being queried is bold.

4.4 Analysis

Type Errors Correction: We respectively analyzed the error cases again generated by **mulrel** model (α = 0.1) on the AQUAINT dataset (totally 727 mentions) and **wnel** model (β = 0.1) on the AIDA-B dataset (totally 4485 mentions). Type error cases specifically refer to the cases where the type of predicted entity is inconsistent with the ground-truth entity's type and doesn't match the corresponding mention context. For the **mulrel** model, there are 21 type errors cases (about 34.2% of the total errors), 17 of which are corrected by our method. And for the **wnel** model, it generates totally 259 type error cases (about 56.8% of the total errors), we correct 54 of them. As listed in Table 2, our entity embeddings can correct a large number of type error cases. We further analyzed the remaining type error cases and find they are mainly due to the insufficient helpful local context and the low prior probability, which are hard to address. We leave these problems to our future work.

Nearest entities: In addition, we compared the closest entity embeddings in Ganea and Hofmann(2017) and our entity representation space as shown in Table 3.When we query the entity "George Washington", the first three most similar entities in our vector space are all of the */person/politician* category, which is consistent with the type of quried entity "George Washington". While, among the result of Ganea and Hofmann(2017), only one entity remains the same type. It can be seen that our entity embeddings are more sensitive to type.

5 Conclusion

This paper proposes a simple yet effective method to encode fine-grained entity type information into the entity embeddings, enabling models to better capture the fine-grained entity type information. We obtain solid improvement compared with the previous models and the detailed experimental analysis shows our method can correct a fair portion of type error cases produced by the baseline model. Moreover, our method doesn't need to modify the EL model architecture, thus can be easily applied to other EL models.

In the future, we will consider injecting the entity information and fine-grained type information into

Proceedings of the 20th China National Conference on Computational Linguistics, pages 1087-1095, Hohhot, China, Augest 13 - 15, 2021.

the latest pre-trained language model (e.g., BERT (Devlin et al., 2018)) to obtain entity representation and further improve the performance of entity linking.

Acknowledgements

This research work is supported by the National Nature Science Foundation of China (Contract 61876198, 61976015, 61976016). The authors are grateful for the valuable comments and suggestions from the anonymous reviewers.

References

Yixin Cao, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Neural collective entity linking. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 675–686, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

Shuang Chen, Jinpeng Wang, Feng Jiang, and Chin-Yew Lin. 2020. Improving entity linking by modeling latent entity type information. *CoRR*, abs/2001.01447.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96, Melbourne, Australia, July. Association for Computational Linguistics.

Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. 2017. Named entity disambiguation for noisy text. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 58–68, Vancouver, Canada, August. Association for Computational Linguistics.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629, Copenhagen, Denmark, September. Association for Computational Linguistics.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2681–2690, Copenhagen, Denmark, September. Association for Computational Linguistics.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 782–792, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Feng Hou, Ruili Wang, Jun He, and Yi Zhou. 2020. Improving entity linking through semantic reinforced entity embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6843–6848, Online, July. Association for Computational Linguistics.

Nevena Lazic, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2015. Plato: A selective context model for entity resolution. *Transactions of the Association for Computational Linguistics*, 3:503–515.

Phong Le and Ivan Titov. 2018. Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1595–1604, Melbourne, Australia, July. Association for Computational Linguistics.

Phong Le and Ivan Titov. 2019. Boosting entity linking performance by leveraging unlabeled documents. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1935–1945, Florence, Italy, July. Association for Computational Linguistics.

Xiao Ling and Daniel S. Weld. 2012. Fine-grained entity recognition. In *In Proc. of the 26th AAAI Conference on Artificial Intelligence*.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460, Florence, Italy, July. Association for Computational Linguistics.

Laurens J P Van Der Maaten. 2014. Accelerating t-sne using tree-based algorithms. *The Journal of Machine Learning Research*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Computer Science*.

Shruti Rijhwani, Jiateng Xie, Graham Neubig, and Jaime Carbonell. 2019. Zero-shot neural transfer for cross-lingual entity linking. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6924–6931.

Özge Sevgili, Alexander Panchenko, and Chris Biemann. 2019. Improving neural entity disambiguation with graph embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 315–322, Florence, Italy, July. Association for Computational Linguistics.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259, Berlin, Germany, August. Association for Computational Linguistics.

Xiyuan Yang, Xiaotao Gu, Sheng Lin, Siliang Tang, Yueting Zhuang, Fei Wu, Zhigang Chen, Guoping Hu, and Xiang Ren. 2019. Learning dynamic context augmentation for global entity linking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 271–281, Hong Kong, China, November. Association for Computational Linguistics.

Ben Zhou, Daniel Khashabi, Chen-Tse Tsai, and Dan Roth. 2018. Zero-shot open entity typing as type-compatible grounding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2065–2076, Brussels, Belgium, October-November. Association for Computational Linguistics.

Shuyan Zhou, Shruti Rijhwani, John Wieting, Jaime Carbonell, and Graham Neubig. 2020. Improving candidate generation for low-resource cross-lingual entity linking. *Transactions of the Association for Computational Linguistics*, 8:109–124.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Abbas Ghaddar and Philippe Langlais. 2018. Transforming Wikipedia into a large-scale fine-grained entity type corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May. European Language Resources Association (ELRA).

Proceedings of the 20th China National Conference on Computational Linguistics, pages 1087-1095, Hohhot, China, Augest 13 - 15, 2021.